M16C/26

Using the M16C/26 Timers in Timer Mode

1.0 Abstract

The following article describes how to use timers A and B as basic timers, referred to as Timer Mode. Timers are useful for updating multiplexed display, scanning inputs, real time clocks, hardware watchdogs, etc.

2.0 Introduction

The Renesas M30262 is a 16-bit MCU based on the M16C/60 series CPU core. The MCU features include up to 64K bytes of Flash ROM, 2K bytes of RAM, and 4K bytes of Virtual EEPROM. The peripheral set includes 10-bit A/D, UARTS, Timers, DMA, and GPIO. The MCU has eight timers that consists of five Timer A's and three Timer B's. All eight timers can operate in Timer Mode.

Timer A also has the following additional modes of operation:

- Event Counter Mode
- PWM Mode
- One-Shot Mode

Timer B has the following additional modes of operation:

- Event Counter Mode
- Pulse Width Measurement Mode

Figure 1 and Figure 2 show the block diagrams for timers A and B. Note that there are some differences between the two timers but both operate similar in Timer Mode. The remainder of this document will focus on setting up timer A0 in Timer Mode.

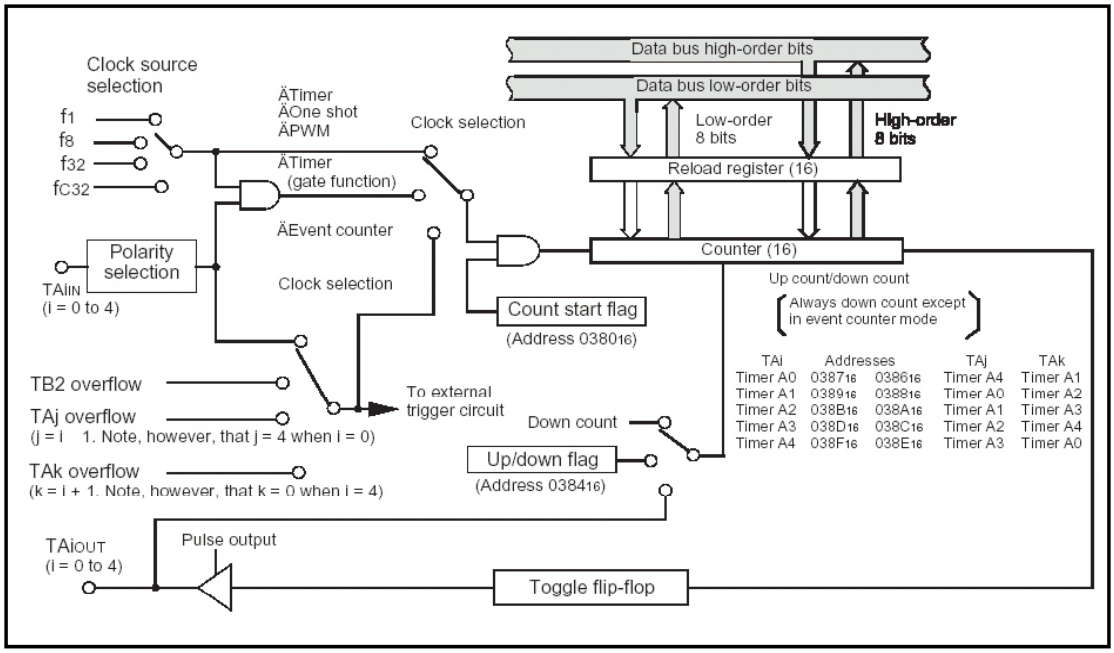


Figure 1 Bock Diagram of Timer A

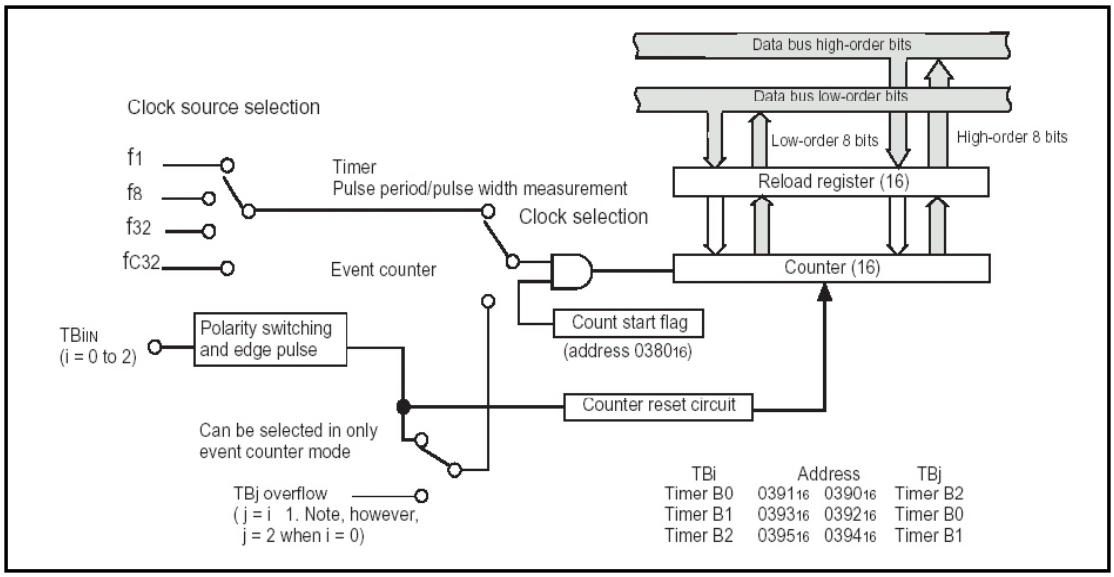


Figure 2 Bock Diagram of Timer A

3.0 Timer Mode Description

In Timer Mode, the counter register counts down using the selected clock source until the counter underflows (0000 to FFFFh). At this point, the value in the reload register is copied into the counter and countdown continues. At the same time, the timer interrupt request bit is set and an interrupt is generated if the timer interrupt priority level is set above the current CPU priority level and the I flag is set. If at any time during countdown the count start flag is cleared, counting is suspended until the start flag is set. Figure 3 illustrates this operation.

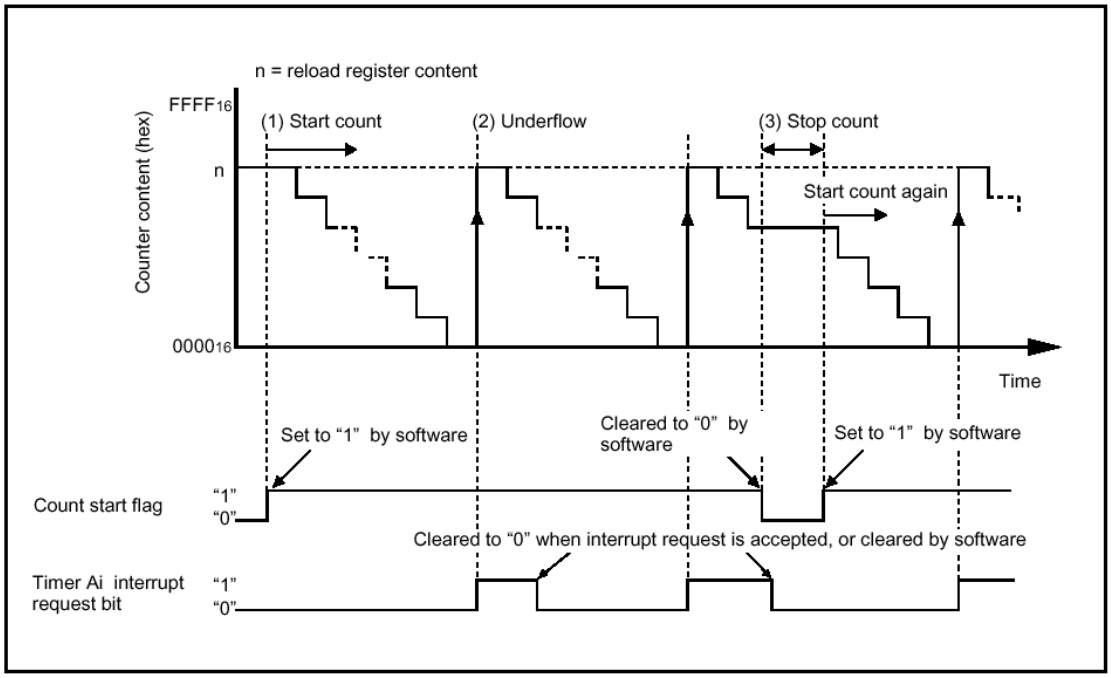


Figure 3 Operation Timing of Timer Mode

4.0 Configuring Timer Mode

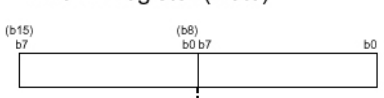
The steps to configure a timer A for Timer Mode are shown below. The steps are similar when setting timer B.

1. Load the TAi register (which also loads the reload register) with the count source
2. Load the timer mode register, TAiMR
   - Select timer mode: bits TMOD0, TMOD1 = 0.
   - Select the clock source (f1, f/8, f/32, or fc/32): bits TCK0, TCK1
3. Load the timer 'interrupt priority level', TAiIC with a value of at least 1
4. Ensure interrupts are enabled (CPU I flag set)
5. Set the 'start count' flag bit, TAiS in the 'count start flag' register, TABSR

It is not necessary to perform these steps in the order listed, however, the count register should be loaded before the 'start count' flag is set. Also, the priority level should not be modified when there is a possibility of an interrupt occurring.

Figure 4 to Figure 7 show the Timer A related registers for Timer mode.

Timer Ai register (Note)



Symbol	Address	When reset
TA0	0387₁₆,0386₁₆	Indeterminate
TA1	0389₁₆,0388₁₆	Indeterminate
TA2	038B₁₆,038A₁₆	Indeterminate
TA3	038D₁₆,038C₁₆	Indeterminate
TA4	038F₁₆,038E₁₆	Indeterminate

Function	Values that can be set	R	W
• Timer mode Counts an internal count source	0000₁₆ to FFFF₁₆	○	○
• Event counter mode Counts pulses from an external source or timer overflow	0000₁₆ to FFFF₁₆	○	○
• One-shot timer mode Counts a one shot width	0000₁₆ to FFFF₁₆	×	○
• Pulse width modulation mode (16-bit PWM) Functions as a 16-bit pulse width modulator	0000₁₆ to FFFE₁₆	×	○
• Pulse width modulation mode (8-bit PWM) Timer low-order address functions as an 8-bit prescaler and high-order address functions as an 8-bit pulse width modulator	00₁₆ to FE₁₆ (Both high-order and low-order addresses)	×	○

Note: Read and write data in 16-bit units.

Figure 4 Timer Ai mode register

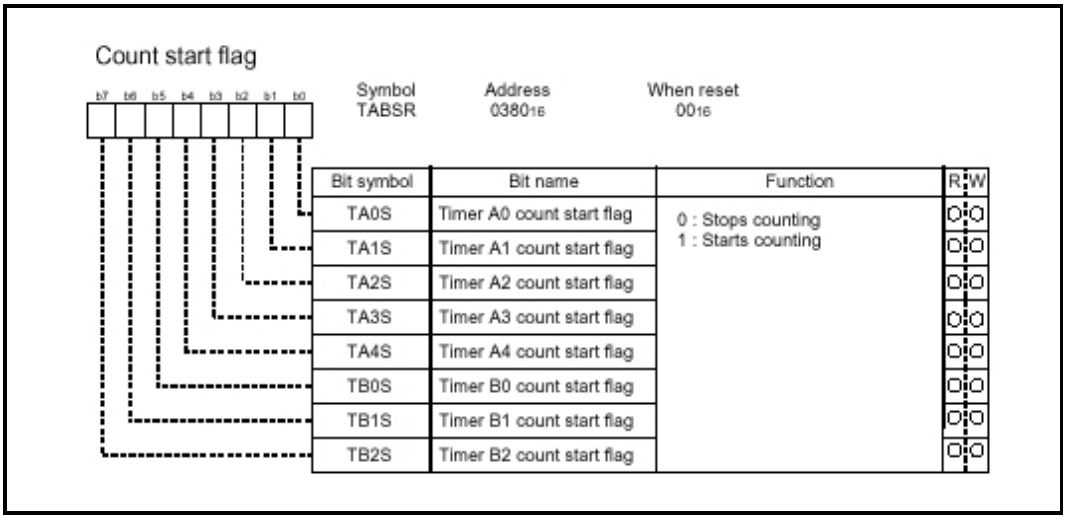
Timer Ai mode register

Symbol	Address	When reset
TAiMR(i=0 to 4)	0396₁₆ to 039A₁₆	00₁₆

Bit symbol	Bit name	Function	R	W
TMOD0	Operation mode select bit	b1 b0 0 0 : Timer mode	○	○
TMOD1			○	○
MR0	Pulse output function select bit	0 : Pulse is not output (TAiOUT pin is a normal port pin) 1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin)	○	○
MR1	Gate function select bit	b4 b3 0 X (Note 2): Gate function not available (TAiIN pin is a normal port pin) 1 0 : Timer counts only when TAiIN pin is held "L" (Note 3) 1 1 : Timer counts only when TAiIN pin is held "H" (Note 3)	○	○
MR2			○	○
MR3	0 (Must always be fixed to "0" in timer mode)		○	○
TCK0	Count source select bit	b7 b6 0 0 : f1 0 1 : f8 1 0 : f32 1 1 : fC32	○	○
TCK1			○	○

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: The bit can be "0" or "1".
Note 3: Set the corresponding port direction register to "0".

Figure 5 Timer Ai mode register



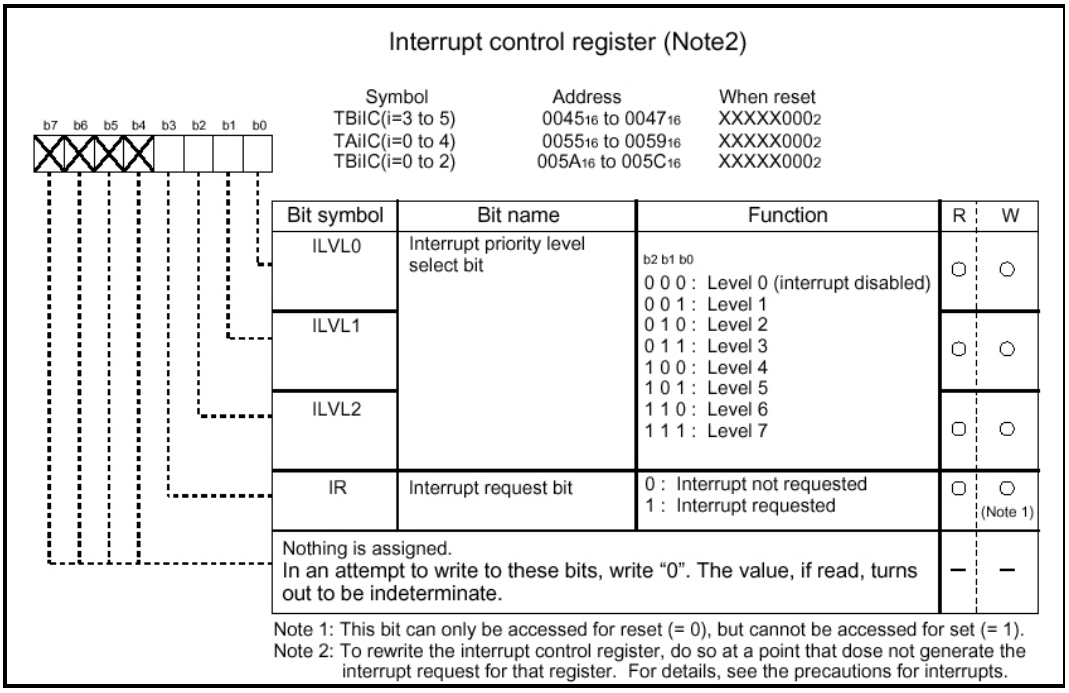
Count start flag

b7	b6	b5	b4	b3	b2	b1	b0

Symbol: TABSR
Address: 0380₁₆
When reset: 00₁₆

Bit symbol	Bit name	Function	R	W
TA0S	Timer A0 count start flag	0 : Stops counting 1 : Starts counting	○	○
TA1S	Timer A1 count start flag		○	○
TA2S	Timer A2 count start flag		○	○
TA3S	Timer A3 count start flag		○	○
TA4S	Timer A4 count start flag		○	○
TB0S	Timer B0 count start flag		○	○
TB1S	Timer B1 count start flag		○	○
TB2S	Timer B2 count start flag		○	○

Figure 6 Count start flag register



Interrupt control register (Note2)

b7	b6	b5	b4	b3	b2	b1	b0
X	X	X	X				

Symbol: TBiIC(i=3 to 5), TAiIC(i=0 to 4), TBiIC(i=0 to 2)
Address: 0045₁₆ to 0047₁₆, 0055₁₆ to 0059₁₆, 005A₁₆ to 005C₁₆
When reset: XXXXX000₂, XXXXX000₂, XXXXX000₂

Bit symbol	Bit name	Function	R	W
ILVL0	Interrupt priority level select bit	b2 b1 b0 0 0 0 : Level 0 (interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	○	○
ILVL1			○	○
ILVL2			○	○
IR	Interrupt request bit	0 : Interrupt not requested 1 : Interrupt requested	○	○ (Note 1)
Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminate.			–	–

Note 1: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).
Note 2: To rewrite the interrupt control register, do so at a point that dose not generate the interrupt request for that register. For details, see the precautions for interrupts.

Figure 7 Interrupt control register

5.0 Reference

Renesas Technology Corporation Semiconductor Home Page

http://www.renesas.com

E-mail Support

techsupport.rta@renesas.com

Data Sheets

- M16C/26 datasheet

User's Manual

- KNC30 Users Manual, KNC30UE.PDF
- M16C/60 and M16C/20 C Language Programming Manual, 6020EC.PDF
- Writing interrupt handlers in C for the M16C Application Note
- MSV30262-SKP or MSV-Mini26-SKP Quick start guide
- MSV30262-SKP or MSV-Mini26-SKP Users Manual
- MDECE30262 or MSV-Mini26-SKP Schematic

6.0 Software Code

A sample program written in C and compiled using the KNC30 compiler to illustrate how to set up a timer mode on timer A0 is listed below. This program runs on the MSV30262 Starter Kit Board and flashes the green LED (D5) at 1 Hz rate.

To become familiar with the timer, try changing the flash rate, the clock source, or even switch to a different timer (e.g. TA1, TB0, etc.).

```
/************************************************************************
*       File Name:    timer_mode.c
*
*       Content:      Example program using Timer A in "Timer Mode" .This program
*              is written for the Timer Mode application note.  This program
*              works with the MSV30262 starter kit board and flashes
*              D5 connected to P7.2.
*
*              Compiled with KNC30.
*
*              All timing based on 20 MHz Xtal
*
*       Copyright 2003 Renesas Technology America, Inc.
*       All Rights Reserved.
*
```

```
*=========================================================================
*      $Log:$
*=========================================================================*/
#include "sfr26.h"

#define  TIME_CONFIG 0x40  /*  01000000 value to load into timer mode register
                                ||||||||_  TMOD0, TMOD1: TIMER MODE SELECTED
                                ||||||____ MR0:          NO PULSE OUTPUT
                                |||||_____ MR1, MR2:     GATE FUNCTION NOT SELECTED
                                |||________ MR3:          SET TO 0 IN TIMER MODE
                                ||_________ TCK0, TCK1:   F8 SELECTED */

#define  CNTR_IPL 0x03   // TA0  priority interrupt level
#define  LED  p7_2       // LED4 is connected to p7_2 on the MSV30262 board

int time_cnt;
int count;               //Global count value, incremented every second

//prototypes
void init(void);

#pragma INTERRUPT /B TimerA0Int
void TimerA0Int(void);

/************************************************************************
Name:         TimerA0Int()
Parameters:   none
Returns:      nothing
Description:  Timer A0 Interrupt Service Routine. Interrupts every 1ms,
              toggles LED every second, and increments global'count'
************************************************************************ */
void TimerA0Int(void)
{
  if ((time_cnt++) > (1000))       // = 1 second
  { LED ^= 1;                      // toggle LED
    count++;                       // example 1 second "clock"
    time_cnt = 0;
  }
}
/************************************************************************
Name:         main()
Parameters:   none
Returns:      nothing
Description: initializes variables and LED port. Then does nothing but
wait for TA0 interrupts.
************************************************************************ */

void main (void)
{
  time_cnt = 0;
  count = 0;
  pd7_2 = 1;
  init();
  while (1);      //LED flashing is interrupt driven
}
```

```
/************************************************************************
Name:         init()
Parameters:   none
Returns:      nothing
Description: Timer TA0 is setup to interrupt every 1ms.
************************************************************************ */
void init()
{
   ta0 = 2500;       // 20MHz xtal divide by 8 at 1ms intervals = 2,500 counts

/* the following procedure for writing an Interrupt Priority Level follows what
   is described in the M16C datasheets under 'Interrupts'. */

   _asm ("   fclr i") ;          //turn off interrupts before modifying IPL
   ta0ic |= CNTR_IPL;          // use read-modify-write instruction to write IPL
   ta0mr = TIME_CONFIG;
   _asm ("   fset i");

   ta0s = 1;  //start counting
}
```

In order for this program to run properly, timer A0's interrupt vector needs to point to the TimerA0Int function. The interrupt vector table is near the end of the startup file "sect30.inc". Insert the function label "_TimerA0Int" into the interrupt vector table at vector 21 as shown below.

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;       File Name:  sect30.inc
;
;       Content:    Section include file for MSV30262-SKP.
;
;       Copyright 2003 Renesas Technology America, Inc.
;       All rights reserved
;
;===========================================================================
;       $Log:$
;===========================================================================
                        :
                        :
       .lword dummy_int             ; uart1 transmit(for user)(vector 19)
       .lword dummy_int             ; uart1 receive(for user)(vector 20)
       .glb   _TimerA0Int
       .lword _TimerA0Int           ; timer A0(for user)(vector 21)
       .lword dummy_int             ; timer A1(for user)(vector 22)
       .lword dummy_int             ; timer A2(for user)(vector 23)
                        :
                        :
```

Keep safety first in your circuit designs!

- Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
  Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
- Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
  The information described here may contain technical inaccuracies or typographical errors.
  Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
  Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (http://www.renesas.com).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
  Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.